

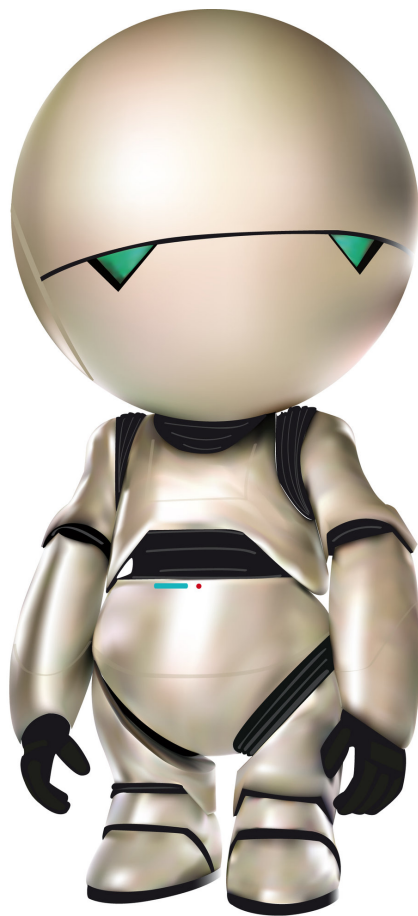




Table des matières

1. Contraintes administratives
2. Rendu
3. Étape 1
4. Bonus



1 Contraintes administratives

- Le projet est à rendre pour le dimanche soir 23h 42min 42s.
- Le sujet est à faire par groupe de 2 personnes.
- La soutenance est *obligatoire*, même si vous n'avez rien fait.
- Vous serez notés par groupe (une note pour tout le groupe).
- Le sujet pourra changer jusqu'à 1H avant le rendu.
- Tout ajout ou complément au sujet est susceptible de se produire et se trouvera dans le forum.
- Vous devez valider votre participation au rush et donc à la soutenance sur l'intra. En cas d'absence la sanction sera -21.
- Vous n'avez le droit qu'à la libc
- TOUTES les étapes sont obligatoires!!!!



2 Rendu

2.1 Généralité

- Vous devez obligatoirement rendre toutes vos sources.

2.2 Répertoire

- Le rendu se fait via *git* sur le compte du chef de groupe.
- Le nom du repo est *marvin*.
- Vous devez mettre les droits en lecture pour le compte rush

2.3 Makefile

Il devra y avoir un Makefile qui contiendra:

- règle *clean* qui effacera tous les objets
- règle *fclean* qui effacera tous les objets et les exécutables
- règle *re* qui procédera à un *fclean* puis à un *all*.
- règle *all* qui compilera le projet.
- Vous devrez obligatoirement compiler sans warning avec les options de compilation : `-W -Wall -ansi -pedantic`.
- La règle par défaut sera *all*. Faites attention à ne pas relinker vos fichiers s'ils n'ont pas été modifiés. Sachez faire un Makefile!

2.4 Fichier auteur

Il devra y avoir un fichier nommé *auteur* qui contiendra les logins de toutes les personnes du groupe, chacun suivi d'un `\n`, tel que:

```
(elbarto@atlantis) cat -e auteur
elbarto$
sha-1$
(elbarto@atlantis)
```



3 Etape 1

Votre programme affichera en français son premier paramètre :

Exemple 1

```
./marvin "int a;"
```

Déclaration de la variable de type entier nommé a.

Exemple 2

```
./marvin "int aff_nbr(int a);"
```

Déclaration de la fonction `aff_nbr` qui prend en paramètre un entier nommé a et renvoyant un entier.

Exemple 3

```
./marvin "int main(int ac, char *av[]);"
```

Déclaration de la fonction `main` prenant en paramètre un entier nommé `ac` et un pointeur sur un pointeur sur caractère nommé `av` et renvoyant un entier.

Exemple 4

```
./marvin "long (*f)(void)(char *str);"
```

Déclaration du pointeur sur fonction `f` qui ne prend rien en paramètre et renvoyant une fonction qui prend en paramètre un pointeur sur caractère nommé `str` et renvoyant un entier `long`.

Exemple 5

```
./marvin "t_ex *eve(void);"
```

Déclaration de la fonction `eve` qui ne prend rien en paramètre et renvoyant un pointeur sur type `t_ex`.



S'il n'y a pas d'argument ou plus d'un à votre executable, affichez \n.

On ne veut voir AUCUN segfault.

On ne veut voir AUCUN appel système sans que sa valeur de retour ne soit testée.

Vous devez gérer les erreurs de syntaxe. Vous devez afficher un message le plus explicite possible.

1. Caractère où l'erreur s'est produite.
2. Type de l'erreur.
3. Etc ...

4 Restrictions

On se limitera exclusivement aux déclarations.



5 Bonus

5.1 Bonus 1

Vous pouvez gérer en bonus des expressions de la forme :

```
int a = 4;
```

ou encore plus dur :

```
int a = blague("toto", 456, 4.5, '\t');
```

ou encore :

```
int a = plouf((float)45, "glouglou");
```

5.2 Bonus 2

Vous pouvez gérer plusieurs langues.

